

TRANSMITTAL LETTER TO THE UNITED STATES

DESIGNATED/ELECTED OFFICE (DO/EO/US)

CONCERNING A FILING UNDER 35 U.S.C. 371

112740-319

U.S. APPLICATION NO. (IF KNOWN, SEE 37 CFR

09/936385

INTERNATIONAL APPLICATION NO.

PCT/DE00/00623

INTERNATIONAL FILING DATE

01 March 2000

PRIORITY DATE CLAIMED

09 March 1999

TITLE OF INVENTION

METHOD FOR ALLOCATION OF A QUALITY OF SERVICE FOR A PACKET STREAM

APPLICANT(S) FOR DO/EO/US

Michael Wagner

Applicant herewith submits to the United States Designated/Elected Office (DO/EO/US) the following items and other information:

1. ☒ This is a **FIRST** submission of items concerning a filing under 35 U.S.C. 371.
2. ☐ This is a **SECOND** or **SUBSEQUENT** submission of items concerning a filing under 35 U.S.C. 371.
3. ☒ This is an express request to begin national examination procedures (35 U.S.C. 371(f)) at any time rather than delay examination until the expiration of the applicable time limit set in 35 U.S.C. 371(b) and PCT Articles 22 and 39(1).
4. ☒ A proper Demand for International Preliminary Examination was made by the 19th month from the earliest claimed priority date.
5. ☒ A copy of the International Application as filed (35 U.S.C. 371 (c) (2))
 - a. ☒ is transmitted herewith (required only if not transmitted by the International Bureau).
 - b. ☐ has been transmitted by the International Bureau.
 - c. ☐ is not required, as the application was filed in the United States Receiving Office (RO/US).
- ☒ A translation of the International Application into English (35 U.S.C. 371(c)(2)).
- ☒ A copy of the International Search Report (PCT/ISA/210).
- ☒ Amendments to the claims of the International Application under PCT Article 19 (35 U.S.C. 371 (c)(3))
 - a. ☒ are transmitted herewith (required only if not transmitted by the International Bureau).
 - b. ☐ have been transmitted by the International Bureau.
 - c. ☐ have not been made; however, the time limit for making such amendments has NOT expired.
 - d. ☐ have not been made and will not be made.
- ☒ A translation of the amendments to the claims under PCT Article 19 (35 U.S.C. 371(c)(3)).
- ☐ An oath or declaration of the inventor(s) (35 U.S.C. 371 (c)(4)).
- ☒ A copy of the International Preliminary Examination Report (PCT/IPEA/409).
- ☐ A translation of the annexes to the International Preliminary Examination Report under PCT Article 36 (35 U.S.C. 371 (c)(5)).

Items 13 to 20 below concern document(s) or information included:

13. ☐ An Information Disclosure Statement under 37 CFR 1.97 and 1.98.
14. ☐ An assignment document for recording. A separate cover sheet in compliance with 37 CFR 3.28 and 3.31 is included.
15. ☒ A **FIRST** preliminary amendment.
16. ☐ A **SECOND** or **SUBSEQUENT** preliminary amendment.
17. ☒ A substitute specification.
18. ☐ A change of power of attorney and/or address letter.
19. ☒ Certificate of Mailing by Express Mail
20. ☒ Other items or information:

Submission of Drawings - Figures 1-2 on two sheets

21. The following fees are submitted:

BASIC NATIONAL FEE (37 CFR 1.492 (a) (1) - (5)) :

- ☐ Neither international preliminary examination fee (37 CFR 1.482) nor international search fee (37 CFR 1.445(a)(2)) paid to USPTO and International Search Report not prepared by the EPO or JPO \$1,000.00
- ☒ International preliminary examination fee (37 CFR 1.482) not paid to USPTO but International Search Report prepared by the EPO or JPO \$860.00
- ☐ International preliminary examination fee (37 CFR 1.482) not paid to USPTO but international search fee (37 CFR 1.445(a)(2)) paid to USPTO \$710.00
- ☐ International preliminary examination fee paid to USPTO (37 CFR 1.482) but all claims did not satisfy provisions of PCT Article 33(1)-(4) \$690.00
- ☐ International preliminary examination fee paid to USPTO (37 CFR 1.482) and all claims satisfied provisions of PCT Article 33(1)-(4) \$100.00

ENTER APPROPRIATE BASIC FEE AMOUNT =

CALCULATIONS PTO USE ONLY

\$860.00

Surcharge of \$130.00 for furnishing the oath or declaration later than ☐ 20 ☐ 30 months from the earliest claimed priority date (37 CFR 1.492 (e)).

\$0.00

CLAIMS	NUMBER FILED	NUMBER EXTRA	RATE
Total claims	5 - 20 =	0	x \$18.00
Independent claims	1 - 3 =	0	x \$80.00
Multiple Dependent Claims (check if applicable).			<input type="checkbox"/>

\$0.00

TOTAL OF ABOVE CALCULATIONS =

\$860.00

Reduction of 1/2 for filing by small entity, if applicable. Verified Small Entity Statement must also be filed (Note 37 CFR 1.9, 1.27, 1.28) (check if applicable).

☐

\$0.00

SUBTOTAL =

\$860.00

Processing fee of \$130.00 for furnishing the English translation later than ☐ 20 ☐ 30 months from the earliest claimed priority date (37 CFR 1.492 (f)).

+

\$0.00

TOTAL NATIONAL FEE =

\$860.00

Fee for recording the enclosed assignment (37 CFR 1.21(h)). The assignment must be accompanied by an appropriate cover sheet (37 CFR 3.28, 3.31) (check if applicable).

☐

\$0.00

TOTAL FEES ENCLOSED =

\$860.00

Amount to be:
refunded \$
charged \$

☒ A check in the amount of **\$860.00** to cover the above fees is enclosed.

☐ Please charge my Deposit Account No. _____ in the amount of _____ to cover the above fees.
A duplicate copy of this sheet is enclosed.

☒ The Commissioner is hereby authorized to charge any fees which may be required, or credit any overpayment to Deposit Account No. **02-1818** A duplicate copy of this sheet is enclosed.

NOTE: Where an appropriate time limit under 37 CFR 1.494 or 1.495 has not been met, a petition to revive (37 CFR 1.137(a) or (b)) must be filed and granted to restore the application to pending status.

SEND ALL CORRESPONDENCE TO:

William E. Vaughan (Reg. No. 39,056)
Bell, Boyd & Lloyd LLC
P.O. Box 1135
Chicago, Illinois 60690

SIGNATURE

William E. Vaughan

NAME

39,056

REGISTRATION NUMBER

September 10, 2001

DATE

00/036385
2000 SEP 20 10 10 SEP 2001

BOX PCT

IN THE UNITED STATES ELECTED/DESIGNATED OFFICE
OF THE UNITED STATES PATENT AND TRADEMARK OFFICE
UNDER THE PATENT COOPERATION TREATY-CHAPTER II

SUBMISSION OF DRAWINGS

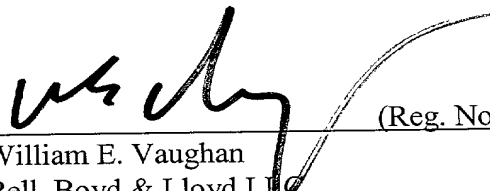
APPLICANT: Michael Wagner DOCKET NO.: 112740-319
SERIAL NO: GROUP ART UNIT:
FILED: EXAMINER:
INTERNATIONAL APPLICATION NO. PCT/DE00/00623
INTERNATIONAL FILING DATE: 01 March 2000
INVENTION: METHOD FOR ALLOCATION OF A QUALITY OF SERVICE FOR
A PACKET STREAM

Assistant Commissioner for Patents,
Washington, D.C. 20231

Sir:

Applicant herewith submits two sheets (Figs. 1-2) of drawings for the above-
referenced PCT application.

Respectfully submitted,


(Reg. No. 39,056)
William E. Vaughan
Bell, Boyd & Lloyd LLC
P.O. Box 1135
Chicago, Illinois 60690-1135
(312) 807-4292
Attorneys for Applicant

BOX PCT

IN THE UNITED STATES ELECTED/DESIGNATED OFFICE
OF THE UNITED STATES PATENT AND TRADEMARK OFFICE
UNDER THE PATENT COOPERATION TREATY-CHAPTER II

5

PRELIMINARY AMENDMENT

APPLICANT: Michael Wagner. DOCKET NO: 112740-319

SERIAL NO: GROUP ART UNIT:

EXAMINER:

INTERNATIONAL APPLICATION NO: PCT/DE00/00623

10 INTERNATIONAL FILING DATE: 01 March 2000

INVENTION: METHOD FOR ALLOCATION OF A QUALITY OF
SERVICE FOR A PACKET STREAM

15 Assistant Commissioner for Patents,
Washington, D.C. 20231

Sir:

Please amend the above-identified International Application before entry
into the National stage before the U.S. Patent and Trademark Office under 35

20 U.S.C. §371 as follows:

In the Specification:

Please replace the Specification of the present application, including the
Abstract, with the following Substitute Specification:

S P E C I F I C A T I O N

25

TITLE OF THE INVENTION

METHOD FOR TRANSMITTING MESSAGES BETWEEN A CLIENT
INSTANCE ASSIGNED TO A FIRST PROCESS AND AT LEAST ONE
SERVER INSTANCE ASSIGNED TO AT LEAST ONE FURTHER PROCESS
WITHIN A DISTRIBUTED SYSTEM

BACKGROUND OF THE INVENTION

Distributed systems preferably play a particular role in contemporary telecommunications systems which are generally multiprocessor systems. A distributed system is characterized, in particular, by the fact that processes can be respectively assigned to different processors, and the processors can, if appropriate, be located at spatially separate platforms in the distributed system. In such a case, one of the most important aspects of the communication between the various processes of a distributed system is the platform transparency. This means that a process which wishes to transmit a message to another process does not need to know the platform on which the other process is running at that particular time. Nowadays, such a complex distributed system must also fulfil a larger number of other requirements. It must, inter alia, prove to be extremely reliable, as flexible as possible and as open as possible to adaptations and expansions. The software of such a complex distributed system therefore must be configured in a highly modular fashion with permanently defined open interfaces to the outside so that the individual modules of software are easily adaptable and particularly re-usable.

In order to be able to comply with the abovementioned requirements, in particular in terms of re-usability of software, the software for such a distributed system is generated using object-oriented design methods and/or interprogramming. However, the allocation, necessary in distributed systems, of objects to one another which are usually assigned to different and possibly concurrently running processes, is not solved to a satisfactory degree. To a certain extent, even a purely object-oriented system design must be broken up into conventional procedural programmer techniques, as a result of which the effect of re-using program parts which is achieved with the object orientation is more or less lost.

At present, the following known approaches are being discussed with regard to the introduction of concurrent running and parallel processing into the world of objects:

- Implicit concurrent running: When implicit concurrent running is implemented, there are two possibilities:

- Passive objects: An asynchronous exchange of messages is converted into a sequential synchronous method call or procedure call. Here, the parallel processing of the objects which communicate with each other is very restricted.
- 5 - Active objects: The process is started for each object. This procedure leads to a high level of consumption of resources and is therefore only capable of being implemented with a limited number of objects.
- Explicit concurrent running: Here, either a group of objects (object-related), as described in an article by A. Coutts, J.M. Edwards, Model-Driven Distributed Systems, IEEE Concurrency, July 1997, pp. 55-63, or a plurality of events in a sequence (task-related), as explained in an article by M. Awad, J. Ziegler, A Practical Approach to the Design of Concurrency in Object-Oriented Systems, Software - Practice and Experience, September 1997, Vol. 27(9), pp. 1013-1034, are assigned to a process. If the right-hand half of Figure 3 in the aforesaid article by Awad/Ziegler and Figure 5 in the aforesaid article by Coutts/Edwards are considered, it is apparent, at the interfaces between the objects, some of which at the same time represent interfaces between the processes, that the communication between the objects is carried out both via synchronous method calls and via interprocess communication in the form of the asynchronous passing on of messages. Such a definition of the method of communication at the interfaces of objects has the disadvantage that it is made considerably more difficult to re-use and maintain the objects.

Particularly in the context of the communication between various objects of a distributed system, also referred to as instances, which as a rule have what is known as a client/server relationship with one another and which are assigned to various processes, the procedure explained above is a very unfavorable solution in terms of the possibilities of re-use and maintenance which are desired in such a complex system.

0936339-16601
TOP SECRET 5336660

A method for converting an interface definition description in an inter-object communications system is already known (EP-A-0 860 776) in which a client object and a server object are provided which are either operated on the same computer or on different computers.

5 The respective known method is based here on the function of, on the one hand, providing a programming method which is concentrated on origin processing, made available by a server object, while the advantages of a CORBA architecture, that is to say an architecture with a common (object request) broker are to be retained or ensured, and on the other hand of making available an inter-object
10 communications method.

 For this purpose there is provision to transmit a message from the client object to the server object in order to execute a specific processing operation. An interface definition conversion program here converts an interface definition description which is written in an interface definition language in order to generate
15 what is referred to as a client stub, a server skeleton and a routing program . The aforementioned interface definition description includes one or more method definition descriptions of data which are necessary for the aforesaid specific processing operation, and a processing result and a message description which specifies a format of the message which is to be output in response to the respective
20 method definition description.

 The aforementioned client stub is called in order to cause the client object to output the message. The server skeleton includes a server registration method for storing a routing information item in a routing information memory in order to specify the format of the respective message which can process the server object
25 itself when it is started. In addition, the aforementioned server skeleton includes a method for describing the aforementioned specific processing operation which is to be executed when the server object receives the message.

 Finally, the aforementioned routing program decides whether or not the processing of the server object assigned to the routing information item of the

aforementioned message is possible, specifically by comparing the aforementioned message with the aforementioned routing information.

It is also known (EP-A-0 623 876) to allow object managers, which are provided on different computer platforms, to communicate with one another in a cooperative fashion while the objects are enabled to communicate on the respective computer platforms using a remote procedure request.

Furthermore, a method and a device for carrying out efficient CORBA transactions are known (EP-A-0 834 807). However, in the method according to the present invention, procedures are not performed using CORBA transactions.

Finally, further methods for setting up connections between a server and a client are also known (US-A-5 802 367, WO 98 02814 A, GB-A-2 305 270) which, however, solve problems other than those solved by the present invention.

An object of the present invention consists, therefore, in configuring a method for transmitting messages between what is referred to as client and server instances of a distributed system which are respectively assigned to different processes, to the effect that in terms of the implementation of the method, the greatest possible degree of re-use is provided and, at the same time, maintenance is made as easy as possible.

SUMMARY OF THE INVENTION

The object specified above is achieved according to the present invention with a method of the type mentioned at the beginning by virtue of the fact that, after reception of a message directed to at least one server instance by the client instance, a first instance (object handler 1), of partner instances provided as mutual communications partners, which contains the first process selects at least one suitable further instance(object handler 2), containing the at least one further process, of the partner instances, to receive and pass on messages with reference to an allocation table, and by virtue of the fact that the respective further instance (object handler 2) passes on the message to at least one server instance addressed by it, and, if appropriate, receives from the at least one server instance a message to be passed on to the client instance via the first instance containing the first process.

The present invention provides the advantage that the definition of the method of communication between the client instance and the at least one server instance is exported into the partner instances which contain a process and which are provided as mutual communications partners. In this way, the messages between the client instance and the first instance containing the first process as well as between the further instance containing at least one server instance and the at least one further process are transmitted synchronously; for example, via a procedure call or method call. The transmission of messages between a first instance containing the first process and a further instance containing at least one further process can then take place asynchronously or synchronously, in a way which is decoupled from the communications interfaces of the client instance and the at least one server instance. As a result, a maximum degree of re-use is achieved especially in terms of the implementation of the client instance and of the at least one server instance. The possibility of maintenance is also considerably improved by virtue of the fact that most communications interfaces between the first instance containing the first process and the further instance containing the at least one further process have to be adapted, but the communications interfaces of the client and of the at least one server instance remain untouched.

A further advantageous embodiment of the present invention provides for the allocation table to contain the type of messages which can be output by the client instance and the address of the further instance containing at least one further process. The type of messages which can be output by the client instance and the address of the further instance which contains at least one further process are therefore entered in this allocation table. This allocation table has the advantage that its contents can be changed at any time and that it is made possible for the first instance containing the first process to make a rapid selection.

According to one embodiment of the present invention, the selection made by the first instance containing the first process can be modified dynamically as a function of the system loading. As a result, the system crashes and blockages in the allocation of the processes to the processors can be avoided.

A further embodiment of the present invention relates to the special case in which the first process and the at least one further process coincide. In this case, the first instance containing the first process and the further instance containing at least one further process are combined in one instance. As a result, the method according to the present invention can be applied to this special case without adaptation.

A further useful embodiment of the present invention can be seen in the method of implementation. For example, all the instances (client instance, server instance, the instance containing the first process and the partner instance) can be implemented in the form of objects whose structure is defined by object classes.

Thus, the first instance containing the first process and the further instance containing the at least one further process preferably each having the structure of a common object class. In this way, the principles of purely object-oriented programming are utilized, permitting a high degree of modularity and of re-use and ease of maintenance.

A further-embodiment of the present invention is the very expedient use of the method in a telephone switching system. According to this, all the advantages mentioned above also can be exploited in conjunction with a telephone switching system.

Additional features and advantages of the present invention are described in, and will be apparent from, the following Detailed Description of the Invention and the Figures.

BRIEF DESCRIPTION OF THE FIGURES

Figure 1 shows an exemplary flowchart of the method according to the present invention.

Figure 2 shows an example of an application in the field of a system alarm in a telecommunications system such as a telephone switching system.

A key to the figures is provided at the end of the Detailed Description.

DETAILED DESCRIPTION OF THE INVENTION

Figure 1 describes, in a flowchart, the transmission of messages between a client instance assigned to a first process and a server instance assigned to a further

process. The instances of client, server, the first instance containing the first process and the further instance containing the at least one further process as well as the action which is carried out by the server instance are represented in the form of objects with boxes. Accordingly, the object client corresponds to a client instance, the object server to a server instance, the object object handler 1 to a first active instance containing the first process, of the partner instances provided as mutual communications partners, the object object handler 2 to a further active instance containing the further process, of the partner instances, the object action to an action and the object confirm action to an acknowledgement action to a requested action. The active instances which contain the respective processes are characterized here via boxes with bold lines. The type of action is not determined until the specific object action is called.

In the case of an action which is requested by the client and is to be carried out by the server, with an acknowledgement, the method proceeds, for example, as follows:

The client requests from the server an action to which an acknowledgement is to be made. The client calls the action and does not need to know which process is to be carried out or on which processor platform the action is to be carried out. The object handler 1 provides the client for this purpose with the call procedure `invoke_action`. After the call of the procedure `invoke_action`, also referred to as method in the object-oriented programming, a uniquely defined number (get handle number) is assigned to the object handler 1 and a timer is started (start timer) which initiates error handling if the acknowledgement is not received at the correct time. Then, the object handler 1 looks for a partner instance provided as communications partner, for example object handler 2 (find target object handler), which is assigned to the action as a function of the type of action, and transmits the message of the action request `action_request` to the object handler 2. The object handler 2 receives the message, stores the address of its communications partner object handler 1 (store communication partner) together with the number which is uniquely assigned to the object handler 1 and executes the procedure of the object action (`execute`).

093633-4601

The object action subsequently causes the server addressed by the client to execute the action via the procedure call action. After the execution of the action, the server transmits, in an analogous fashion, an acknowledgement indirectly back to the client. According to this, the following procedure calls, message transmissions and actions run from the server in the direction of the client. The procedure call invoke_action, deletes the address of the communications partner and transmits the action request message for the acknowledgement action_request of the object handler 2 to the object handler 1, which is known to the object handler 2 on the basis of the assigned number, the object handler 1 deletes the assigned number (release handle number) and stops the timer (stop timer), in order to transmit the acknowledgement object handler 1 calls the procedure execute of the object Confirm action, and finally Object Confirm Action executes the procedure confirm_action of the client.

In the case of an action of the server requested by the client being without an acknowledgement, the method according to the present invention of the transmission of messages from the client to the server runs in a similar fashion to that described above. The sequence steps get handle number, start timer, store communication partner and the steps relating to the acknowledgement from the server in the direction of the client are eliminated.

In the case of what is referred to as a broadcast, i.e. a client requests an action from a number of servers, there are various possibilities:

- if the servers addressed by the client are assigned to a common process, the object handler 1 will pass on the action_request message either to an object handler 2, and the object handler 2 ensures that the action is executed by a number of servers, or the object handler 1 transmits a number of action_request messages to a number of object handlers 2 containing the server process, which each cause the servers to execute the action. A combination of both aforesaid variants is also possible.
- if the servers addressed by the client are assigned to different processes, the object handler 1 will in each case transmit an action_request message to the

object handlers 2 containing the different processes, and the object handlers 2 in each case cause the servers to execute the action.

Here too, all combinations of the aforesaid possibilities are conceivable.

The number of actions usually have to be carried out in a distributed system
5 so that each server also can act as a client and each client also can act as a server, and can be combined in an object client and server function.

The advantageous decoupling of process interfaces from the object
interfaces of the client and of the server is apparent from the fact that the
communication between the client and the server is implemented synchronously via
10 procedure calls and method calls and only the passing on of messages between the object handler 1 and the object handler 2 is, if appropriate, carried out asynchronously without limitation to the process limits.

In the special case in which the client and the server, which are located, for
example, on a common platform, can be assigned to the same process, the objects
15 object handler 1 and object handler 2 are combined to form a single object. According to Figure 1, the object handler 1 transmits the action_request message to itself.

Figure 2 shows an application example in the field of a system alarm in a telecommunications system; for example, a telephone switching system.

20 In a system alarm, there are, for example, the following objects which can act simultaneously as client and server and can request different actions from one another. In addition, the objects can be located on different platforms.

An object Alarm Balance Monitor (ABM) has the function of forming an alarm balance over all the alarms of the instances (AMOI) which are monitored by
25 it and for which alarms can be given. In order to be able to form the alarm balance, the Alarm Balance Monitor requires what is referred to as a SIBS object which is located on a processor platform and provides it with collected information relating to the monitored instances for which alarms can be given.

The boxes constitute the objects Caller, AMOI
30 (AlarmManagedObjectInstance), SIBS (SiteBalanceSupply) and ABM

(AlarmBalanceMonitor). The arrows whose type is not given in the key in the index indicate the message transmission, if appropriate, without limitation to process boundaries, between the objects. The transmission of messages correspond here to the transmission of messages between client and server described in Figure 1. Thus, for example, the Caller object can act as a client and the AMOI object as a server. The same also applies to the other objects AMOI and SIBS as well as SIBS and ABM.

After a system alarm call `set_alarm`, the following sequence of actions is triggered, for example:

- 10 - `Set_alarm`: a monitored instance AMOI for which an alarm can be given receives a new alarm from a caller, checks the parameters (`check_params`) determining the alarm and creates a new alarm instance (`create contained alarm`).
- `Confirm`: an acknowledgement from the instance (AMOI) to the instance Caller after the system alarm call `set_alarm`.
- 15 - `Balance SIBS`: at least one server object SIBS is requested to collect the received information necessary for the alarm balance (`accumulate alarm status of all associated AMOI`).
- `Balance ABM`: after this the server object ABM is requested to collect the information received from the at least one SIBS object for the alarm balance (accumulate alarm status of all associated SIBS).
- 20

Because the actions are requested without limitation to process boundaries, the messages are transmitted from one object to a further object via an active first instance and via a further active partner instance, for example via the object handler 1 and via the object handler 2 from Figure 1, neither of which is illustrated in Figure 2.

The selection of the object handler 2 made by the object handler 1 can be performed via an allocation table. The allocation table looks, for example, as follows:

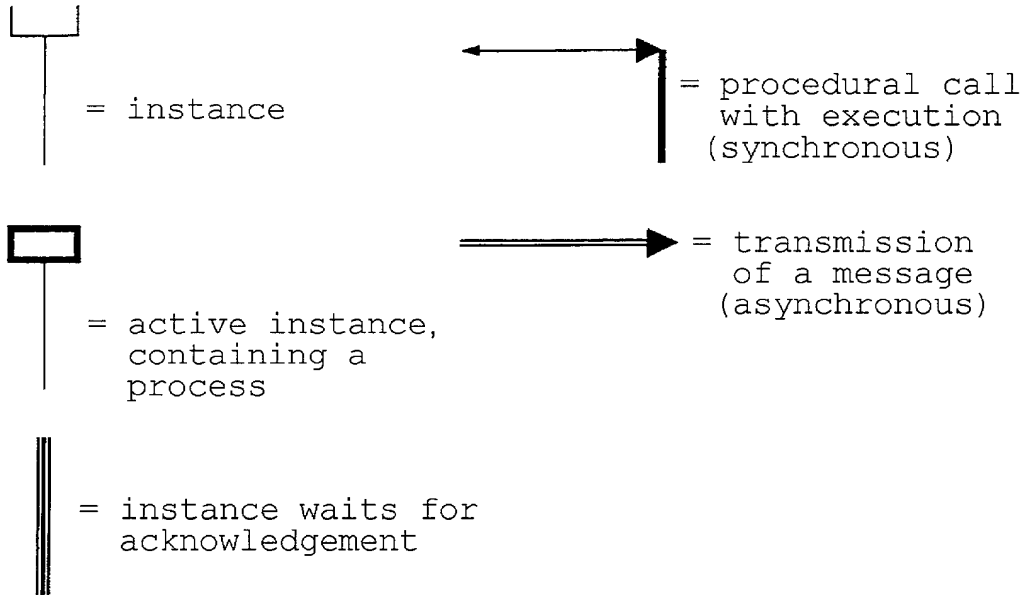
30

Action	Object handler	Confirmation
Set_alarm AMOI	on AMOI platform	yes
Balance SIBS	on SIBS platform	no
Balance ABM	on main platform	no

If a specific action can be executed by different server objects, the allocation of the object handler 2 can be modified as a function of the system load factor.

- 5 Although the present invention has been described with reference to specific embodiments, those of skill in the art will recognize that changes may be made thereto without departing from the spirit and scope of the invention as set forth in the hereafter appended claims.

Key to the figures:



ABSTRACT OF THE DISCLOSURE

A method for transmitting messages between a client instance assigned to a first process and at least one server instance assigned to at least one further process within a distributed system, wherein a first instance containing a first process, of partner instances provided as mutual communications partners, selects, after reception of a message directed to at least one server instance by the client instance at least one suitable further instance containing the at least one further process, of the partner instances for the reception and passing on of messages. The at least one further instance containing the at least one further process passes on this message to at least one server instance addressed by it and receives, if appropriate, from the at least server instance, a message to be passed on to the client instance via the first instance containing the first process.

In the Claims:

On page 13, cancel line 1, and substitute the following left-hand justified heading therefor:

CLAIMS

Please cancel claims 1-6, without prejudice, and substitute the following claims therefor:

7. A method for transmitting messages between a client instance assigned to a first process and at least one server instance assigned to a further process within a distributed system, the method comprising the steps of:
 - receiving a message directed from the client instance to the at least one server instance;
 - selecting, via a first instance containing the first process, from partner instances provided as mutual communications partners, at least one suitable further instance, containing the further process, of the partner instances for receiving and passing on of messages via an allocation table between a type of messages which can be output by the client instance and an address of the further instance which contains at least one further process;

passing on a message, via the respective further instance, to at least one server instance addressed by it; and

receiving, if appropriate, and at the respective further instance, from the at least one server instance a message to be passed on to the client instance via the
5 first instance containing the first process.

8. A method for transmitting messages as claimed in claim 7, the method further comprising the step of:

modifying dynamically the selection made by the first instance containing
10 the first process as a function of a system load factor.

9. A method for transmitting messages as claimed in claim 7, the method further comprising the step of:

combining in one instance the first instance containing the first process and
15 the further instance containing the at least one further process if the first process and the at least one further process coincide.

10. A method for transmitting messages as claimed in claim 7, wherein all of the instances are objects whose structure is defined by object classes.

20

11. A method for transmitting messages as claimed in claim 7, wherein the method is applied to a telephone switching system.

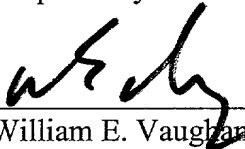
REMARKS

The present amendment makes editorial changes and corrects typographical
25 errors in the specification, which includes the Abstract, in order to conform the specification to the requirements of United States Patent Practice. No new matter is added thereby. Attached hereto is a marked-up version of the changes made to the specification by the present amendment. The attached page is captioned "**Version With Markings To Show Changes Made**".

In addition, the present amendment cancels original claims 1-6 in favor of new claims 7-11. Claims 7-11 have been presented solely because the revisions by crossing out and underlining which would have been necessary in claims 1-6 in order to present those claims in accordance with preferred United States Patent Practice would have been too extensive, and thus would have been too burdensome. The present amendment is intended for clarification purposes only and not for substantial reasons related to patentability pursuant to 35 U.S.C. §§103, 102, 103 or 112. Indeed, the cancellation of claims 1-6 does not constitute an intent on the part of the Applicants to surrender any of the subject matter of claims 1-6.

Early consideration on the merits is respectfully requested.

Respectfully submitted,



(Reg. No. 39,056)
William E. Vaughan
Bell, Boyd & Lloyd LLC
P.O. Box 1135
Chicago, Illinois 60690-1135
(312) 807-4292
Attorneys for Applicant

21.02.2001
9901371

PCT/DE00/00623

Description

Method for transmitting messages between a client instance assigned to a first process and at least one server instance assigned to at least one further process within a distributed system

The invention relates to a method for transmitting messages between a client instance (client) assigned to a first process and at least one server instance (server) assigned to at least one further process within a distributed system.

Distributed systems preferably play a particular role in contemporary telecommunications systems which are generally multiprocessor systems. A distributed system is characterized in particular by the fact that processes can be respectively assigned to different processors, and the processors can, if appropriate, be located at spatially separate platforms in the distributed system. In such a case, one of the most important aspects of the communication between the various processes of a distributed system is the platform transparency. This means that a process which wishes to transmit a message to another process does not need to know the platform on which the other process is running at that particular time. Nowadays, such a complex distributed system must also fulfil a larger number of other requirements. It must, inter alia, prove to be extremely reliable, as flexible as possible and as open as possible to adaptations and expansions. The software of such a complex distributed system must therefore be configured in a highly modular fashion with permanently defined open interfaces to the outside so that the individual modules of software are easily adaptable and particularly re-usable.

In order to be able to comply with the abovementioned requirements, in particular in terms of re-usability of software,

the software for such a distributed system is generated using object-oriented design methods and/or interprogramming. However, the allocation, necessary in distributed systems, of objects to one another which
5 are usually assigned to different and possibly concurrently running processes, is not solved to a satisfactory degree. To a certain extent, even a purely object-oriented system design must be broken up into conventional procedural programmer techniques, as a
10 result of which the effect of re-using program parts which is achieved with the object orientation is more or less lost.

At present, the following known approaches are being
15 discussed with regard to the introduction of concurrent running and parallel processing into the world of objects:

- Implicit concurrent running: When implicit concurrent
20 running is implemented, there are two possibilities:
 - Passive objects: An asynchronous exchange of messages is converted into a sequential synchronous method call or procedure call.
25 Here, the parallel processing of the objects which communicate with each other is very restricted.
 - Active objects: The process is started for each object. This procedure leads to a high
30 level of consumption of resources and is therefore only capable of being implemented with a limited number of objects.
- Explicit concurrent running: Here either a group of
35 objects (object-related), as described in an article by A. Coutts, J.M. Edwards, Model-Driven Distributed Systems, IEEE Concurrency, July 1997, pp. 55-63, or a plurality of events in a sequence (task-related),

21.02.2001
9901371

PCT/DE00/00623
- 2a -

as explained in an article by M. Awad, J. Ziegler, A
Practical Approach to the Design of Concurrency in
Object-Oriented Systems, Software - Practice and
Experience,

September 1997, Vol. 27(9), pp. 1013-1034, are assigned to a process. If the right-hand half of figure 3 in the aforesaid article by Awad/Ziegler and figure 5 in the aforesaid article by Coutts/Edwards are considered, it is apparent, at the interfaces between the objects, some of which at the same time represent interfaces between the processes, that the communication between the objects is carried out both by means of synchronous method calls and by means of interprocess communication in the form of the asynchronous passing on of messages. Such a definition of the method of communication at the interfaces of objects has the disadvantage that it is made considerably more difficult to re-use and maintain the objects.

In particular in the context of the communication between various objects of a distributed system, also referred to as instances, which as a rule have what is known as a client/server relationship with one another and which are assigned to various processes, the procedure explained above is a very unfavorable solution in terms of the possibilities of re-use and maintenance which are desired in such a complex system.

A method for converting an interface definition description in an inter-object communications system is already known (EP-A-0 860 776) in which a client object and a server object are provided which are either operated on the same computer or on different computers.

The respective known method is based here on the function of, on the one hand, providing a programming method which is concentrated on origin processing, made available by a server object, while the advantages of a CORBA architecture, that is to say an architecture

21.02.2001
9901371

PCT/DE00/00623
- 3a -

with a common (object request) broker are to be
retained

or ensured, and on the other hand of making available an inter-object communications method.

For this purpose there is provision to transmit a
5 message from the client object to the server object in
order to execute a specific processing operation. An
interface definition conversion program here converts
an interface definition description which is written in
an interface definition language in order to generate
10 what is referred to as a client stub, a server skeleton
and a routing program. The aforementioned interface
definition description comprises one or more method
definition descriptions of data which are necessary for
the aforesaid specific processing operation, and a
15 processing result and a message description which
specifies a format of the message which is to be output
in response to the respective method definition
description.

20 The aforementioned client stub is called in order to
cause the client object to output the message. The
server skeleton comprises a server registration method
for storing a routing information item in a routing
information memory in order to specify the format of
25 the respective message which can process the server
object itself when it is started. In addition, the
aforementioned server skeleton comprises a method for
describing the aforementioned specific processing
operation which is to be executed when the server
30 object receives the message.

Finally, the aforementioned routing program decides
whether or not the processing of the server object
assigned to the routing information item of the
35 aforementioned message is possible, specifically by
comparing the aforementioned message with the
aforementioned routing information.

21.02.2001
9901371

PCT/DE00/00623
- 4a -

The method according to the present invention cannot be compared with this known procedure.

It is also known (EP-A-0 623 876) to allow object managers, which are provided on different computer platforms, to communicate with one another in a cooperative fashion while the objects are enabled to communicate on the respective computer platforms using a remote procedure request. This procedure also has nothing to do with the method according to the invention.

Furthermore, a method and a device for carrying out efficient CORBA transactions are known (EP-A-0 834 807). However, in the method according to the invention procedures are not performed using CORBA transactions.

Finally, further methods for setting up connections between a server and a client are also known (US-A-5 802 367, WO 98 02814 A, GB-A-2 305 270), which, however, solve problems other than those solved by the present invention.

The object of the invention consists therefore in configuring a method for transmitting messages between what is referred to as client and server instances of a distributed system which are respectively assigned to different processes, to the effect that in terms of the implementation of the method, the greatest possible degree of re-use is provided and at the same time maintenance is made as easy as possible.

The object specified above is achieved according to the invention with a method of the type mentioned at the beginning by virtue of the fact that, after reception of a message directed to at least one server instance by the client instance, a first instance (object handler 1), of partner instances provided as mutual communications partners, which contains the first process selects at least one suitable further instance

(object handler 2), containing the at least one further process, of the partner instances, to receive and pass on messages with reference to an allocation table, and by virtue of the fact that the respective further
5 instance (object handler 2) passes on the message to at least one server instance addressed by it, and, if appropriate, receives from the at least one server instance a message to be passed on to the client instance via the first instance containing the first
10 process.

The invention provides the advantage that the definition of the method of communication between the client instance and the at least one server instance is
15 exported into the partner instances which contain a process and which are provided as mutual communications partners. In this way, the messages between the client instance and the first instance containing the first process as well as between the further instance
20 containing at least one server instance and the at least one further process are transmitted synchronously, for example by means of a procedure call or method call. The transmission of messages between a first instance containing the first process and a
25 further instance containing at least one further process can then take place asynchronously or synchronously, in a way which is decoupled from the communications interfaces of the client instance and the at least one server instance. As a result, a
30 maximum degree of re-use is achieved especially in terms of the implementation of the client instance and of the at least one server instance. The possibility of maintenance is also considerably improved by virtue of the fact that most communications interfaces between
35 the first instance containing the first process and the further instance containing the at least one further process have to be adapted, but the communications

interfaces of the client and of the at least one server instance remain untouched.

5 A further advantageous refinement of the invention provides for the allocation table to contain the type of messages which can be output by the client instance and the address of the further instance containing at least one further process. The type of messages which
10 the further instance which contains at least one further process are therefore entered in this allocation table. This allocation table has the advantage that its contents can be changed at any time and that it is made possible for the first instance
15 containing the first process to make a rapid selection.

According to one expedient development of the invention, the selection made by the first instance containing the first process can be modified
20 dynamically as a function of the system loading. As a result, the system crashes and blockages in the allocation of the processes to the processors can be avoided.

25 A further refinement of the invention relates to the special case in which the first process and the at least one further process coincide. In this case, the first instance containing the first process and the further instance containing at least one further
30 process are combined in one instance. As a result, the method according to the invention can be applied to this special case without adaptation.

35 A further useful refinement of the invention can be seen in the method of implementation. For example, all the instances (client instance, server instance, the instance containing the first process and the partner instance) can be implemented in the form of objects whose

object class. In this way, the principles of purely object-oriented programming are utilized, permitting a high degree of modularity and of re-use and ease of maintenance.

5

A further refinement of the invention is the very expedient use of the method according to the invention in a telephone switching system. According to this, all the advantages mentioned above can also be exploited in
10 conjunction with a telephone switching system.

An exemplary embodiment of the invention is described in more detail below with reference to a drawing, in which:

15

Figure 1 shows an exemplary flowchart of the method according to the invention,

Figure 2 shows an example of an application in the
20 field of a system alarm in a telecommunications system such as a telephone switching system.

A key to the figures is provided in an annex at the end
25 of the description.

Figure 1 describes, in a flowchart, the transmission of messages between a client instance assigned to a first process and a server instance assigned to a further process. The instances of client, server, the first
30 instance containing the first process and the further instance containing the at least one further process as well as the action which is carried out by the server instance are represented in the form of objects with
35 boxes. Accordingly, the object client corresponds to a client instance, the object server to a server instance, the object object handler 1 to a first active instance containing the first process, of the

partner instances provided as mutual communications partners, the object object handler 2 to a further active instance containing the further process, of the partner instances, the object action to an action and
5 the object confirm action to an acknowledgement action to a requested action. The active instances which contain the respective processes are characterized here by means of boxes with bold lines. The type of action is not determined until the specific object action is
10 called.

In the case of an action which is requested by the client and is to be carried out by the server, with an acknowledgement, the method proceeds, for example, as
15 follows:

The client requests from the server an action to which an acknowledgement is to be made. The client calls the action and does not need to know which process is to be carried out or on which processor platform the action is to be carried out. The object handler 1 provides the client for this purpose with the call procedure `invoke_action`. After the call of the procedure `invoke_action`, also referred to as method in the
20 object-oriented programming, a uniquely defined number (get handle number) is assigned to the object handler 1 and a timer is started (start timer) which initiates error handling if the acknowledgement is not received at the correct time. Then, the object handler 1 looks
25 for a partner instance provided as communications partner, for example object handler 2 (find target object handler), which is assigned to the action as a function of the type of action, and transmits the message of the action request `action_request` to the
30 object handler 2. The object handler 2 receives the message, stores the address of its communications partner object handler 1 (store communication partner) together with the number which is uniquely assigned to the object handler 1 and executes the procedure
35

of the object action (execute). The object action subsequently causes the server addressed by the client to execute the action by means of the procedure call action. After the execution of the action, the

server transmits, in an analogous fashion, an acknowledgement indirectly back to the client. According to this, the following procedure calls, message transmissions and actions run from the server in the direction of the client. The procedure call invoke_action, deletes the address of the communications partner and transmits the action request message for the acknowledgement action_request of the object handler 2 to the object handler 1, which is known to the object handler 2 on the basis of the assigned number, the object handler 1 deletes the assigned number (release handle number) and stops the timer (stop timer), in order to transmit the acknowledgement object handler 1 calls the procedure execute of the object Confirm action, and finally Object Confirm Action executes the procedure confirm_action of the client.

In the case of an action of the server requested by the client being without an acknowledgement, the method according to the invention of the transmission of messages from the client to the server runs in a similar fashion to that described above. The sequence steps get handle number, start timer, store communication partner and the steps relating to the acknowledgement from the server in the direction of the client are eliminated.

In the case of what is referred to as a broadcast, i.e. a client requests an action from a plurality of servers, there are various possibilities:

- if the servers addressed by the client are assigned to a common process, the object handler 1 will pass on the action_request message either to an object handler 2, and the object handler 2 ensures that the action is executed by a plurality of servers, or the object handler 1 transmits a plurality of action_request messages to a plurality of object handlers 2 containing the

server process, which each cause the servers to execute the action. A combination of both aforesaid variants is also possible.

- ```

- if the servers addressed by the client are
5 assigned to different processes, the object
 handler 1 will in each case

```

transmit an action\_request message to the object handlers 2 containing the different processes, and the object handlers 2 in each case cause the servers to execute the action.

5

Here too, all combinations of the aforesaid possibilities are conceivable.

10 The plurality of actions usually have to be carried out in a distributed system so that of course each server can also act as a client and each client can also act as a server, and can be combined in an object client and server function.

15 The advantageous decoupling of process interfaces from the object interfaces of the client and of the server is apparent from the fact that the communication between the client and the server is implemented synchronously by means of procedure calls and method  
20 calls and only the passing on of messages between the object handler 1 and the object handler 2 is, if appropriate, carried out asynchronously without limitation to the process limits.

25 In the special case in which the client and the server which are located, for example, on a common platform, can be assigned to the same process, the objects object handler 1 and object handler 2 are combined to form a single object. According to Figure 1, in this case the  
30 object handler 1 transmits the action\_request message to itself.

Figure 2 shows an application example in the field of a system alarm in a telecommunications system, for  
35 example a telephone switching system.

In a system alarm, there are, for example, the following objects which can act simultaneously as client and server and can request different actions

0998385-12601

from one another. In addition,

11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215  
2216  
2217  
2218  
2219  
2220  
2221  
2222  
2223  
2224  
2225  
2226  
2227  
2228  
2229  
223

the objects can be located on different platforms.

An object Alarm Balance Monitor (ABM) has the function of forming an alarm balance over all the alarms of the instances (AMOI) which are monitored by it and for which alarms can be given. In order to be able to form the alarm balance, the Alarm Balance Monitor requires what is referred to as a SIBS object which is located on a processor platform and provides it with collected information relating to the monitored instances for which alarms can be given.

The boxes constitute the objects Caller, AMOI (AlarmManagedObjectInstance), SIBS (SiteBalanceSupply) and ABM (AlarmBalanceMonitor). The arrows whose type is not given in the key in the index indicate the message transmission, if appropriate, without limitation to process boundaries, between the objects. The transmission of messages correspond here to the transmission of messages between client and server described in Figure 1. Thus, for example the Caller object can act as a client and the AMOI object as a server. The same also applies to the other objects AMOI and SIBS as well as SIBS and ABM.

After a system alarm call set\_alarm, the following sequence of actions is triggered, for example:

- Set\_alarm: a monitored instance AMOI for which an alarm can be given receives a new alarm from a caller, checks the parameters (check\_params) determining the alarm and creates a new alarm instance (create contained alarm).
- Confirm: an acknowledgement from the instance (AMOI) to the instance Caller after the system alarm call set\_alarm.
- Balance SIBS: at least one server object SIBS is requested to collect the received information necessary for the alarm balance (accumulate alarm status of all associated AMOI).

- Balance ABM: after this the server object ABM is requested

2025-11-11 14:00:00

to collect the information received from the at least one SIBS object for the alarm balance (accumulate alarm status of all associated SIBS).

- 5 Because the actions are requested without limitation to process boundaries, the messages are transmitted from one object to a further object via an active first instance and via a further active partner instance, for example via the object handler 1 and via the object handler 2 from Figure 1, neither of which is  
10 illustrated in Figure 2.

The selection of the object handler 2 made by the object handler 1 can be performed by means of an  
15 allocation table. The allocation table looks, for example, as follows:

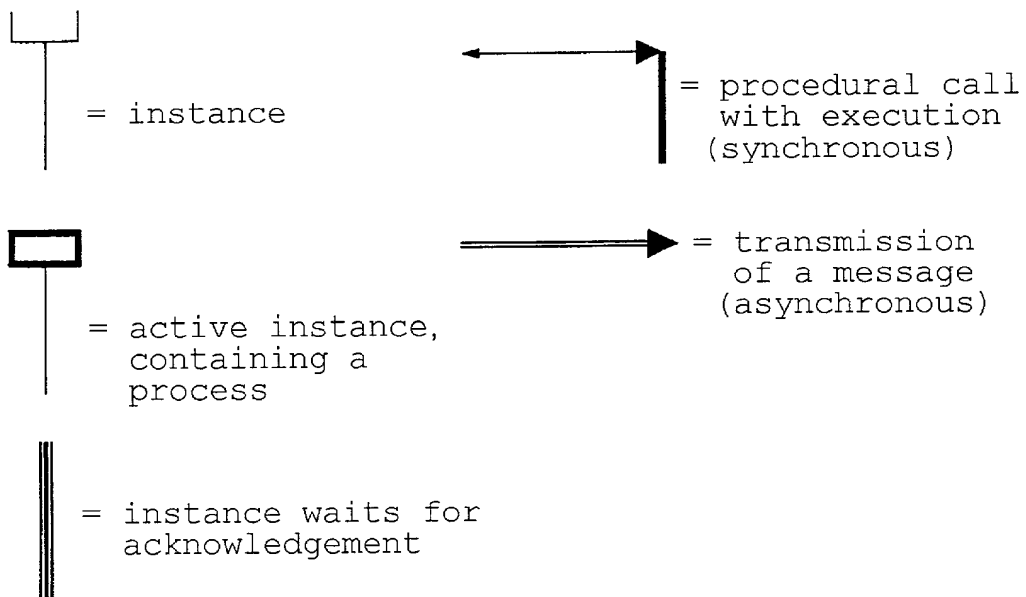
| Action            | Object handler   | Confirmation |
|-------------------|------------------|--------------|
| Set_alarm<br>AMOI | on AMOI platform | yes          |
| Balance<br>SIBS   | on SIBS platform | no           |
| Balance<br>ABM    | on main platform | no           |

If a specific action can be executed by different  
20 server objects, the allocation of the object handler 2 can be modified as a function of the system load factor.

## APPENDIX

Key to the figures:

5





## New Patent claims

(replace the previously applicable patent claims)

1. A method for transmitting messages between a client  
5 instance (client) assigned to a first process and at  
least one server instance (server) assigned to a  
further process within a distributed system,  
characterized in that, after reception of a message  
10 directed from the client instance to at least one  
server instance, a first instance (object handler 1)  
containing the first process selects, from partner  
instances provided as mutual communications partners,  
at least one suitable further instance (object  
15 handler 2), containing the at least one further  
process, of the partner instances for the reception  
and passing on of messages by means of an allocation  
table between the type of messages which can be  
output by the client instance and the address of the  
20 further instance and the address of the further  
instance containing at least one further process, and  
in that the respective further instance (object  
handler 2) passes on the message to at least one  
server instance addressed by it, and if appropriate,  
25 receives from the at least one server instance a  
message to be passed on to the client instance via  
the first instance containing the first process.
2. The method as claimed in claim 1, characterized in  
that the selection made by the first instance  
30 containing the first process can be modified  
dynamically as a function of the system load  
factor.
3. The method as claimed in one of the preceding  
35 claims, characterized in that if the first process  
and the at least one further process coincide, the  
first instance containing the first process and

the further instance containing the at least one  
further process are combined in one instance.

0993884 1999

4. The method as claimed in one of the preceding claims, characterized in that all the instances are implemented in the form of objects whose structure is defined by object classes.

5

5. The method as claimed in one of the preceding claims, characterized in that said method is applied to a telephone switching system.

Description

## SPECIFICATION

### TITLE OF THE INVENTION

5     METHOD FOR TRANSMITTING MESSAGES BETWEEN A CLIENT  
   INSTANCE ASSIGNED TO A FIRST PROCESS AND AT LEAST ONE  
   SERVER INSTANCE ASSIGNED TO AT LEAST ONE FURTHER PROCESS  
          WITHIN A DISTRIBUTED SYSTEM

Method for transmitting messages between a client instance assigned to a first  
process and at least one server instance assigned to at least one further process  
10 within a distributed system  
The invention relates to a method for transmitting messages between a client instance (client)  
assigned to a first process and at least one server instance (server) assigned to at least one further  
process within a distributed system.

### BACKGROUND OF THE INVENTION

15     Distributed systems preferably play a particular role in contemporary  
telecommunications systems which are generally multiprocessor systems. A  
distributed system is characterized, in particular, by the fact that processes can be  
respectively assigned to different processors, and the processors can, if appropriate,  
be located at spatially separate platforms in the distributed system. In such a case,  
20 one of the most important aspects of the communication between the various  
processes of a distributed system is the platform transparency. This means that a  
process which wishes to transmit a message to another process does not need to  
know the platform on which the other process is running at that particular time.  
Nowadays, such a complex distributed system must also fulfil a larger number of  
25 other requirements. It must, inter alia, prove to be extremely reliable, as flexible as  
possible and as open as possible to adaptations and expansions. The software of  
such a complex distributed system must therefore must be configured in a highly  
modular fashion with permanently defined open interfaces to the outside so that the  
individual modules of software are easily adaptable and particularly re-usable.

In order to be able to comply with the abovementioned requirements, in particular in terms of re-usability of software, the software for such a distributed system is generated using object-oriented design methods and/or interprogramming. However, the allocation, necessary in distributed systems, of objects to one another which are usually  
5 assigned to different and possibly concurrently running processes, is not solved to a satisfactory degree. To a certain extent, even a purely object-oriented system design must be broken up into conventional procedural programmer techniques, as a result of which the effect of re-using program parts which is achieved with the object orientation is more or less lost.

10 At present, the following known approaches are being discussed with regard to the introduction of concurrent running and parallel processing into the world of objects:

- Implicit concurrent running: When implicit concurrent running is implemented, there are two possibilities:  
15 - Passive objects: An asynchronous exchange of messages is converted into a sequential synchronous method call or procedure call. Here, the parallel processing of the objects which communicate with each other is very restricted.  
- Active objects: The process is started for each object. This procedure  
20 leads to a high level of consumption of resources and is therefore only capable of being implemented with a limited number of objects.
- Explicit concurrent running: Here, either a group of objects (object-related), as described in an article by A. Coutts, J.M. Edwards, Model-Driven Distributed Systems, IEEE Concurrency, July 1997, pp. 55-63, or a plurality of events in a  
25 sequence (task-related), as explained in an article by M. Awad, J. Ziegler, A Practical Approach to the Design of Concurrency in Object-Oriented Systems, Software - Practice and Experience, September 1997, Vol. 27(9), pp. 1013-1034, are assigned to a process. If the right-hand half of figure Figure 3 in the aforesaid article by Awad/Ziegler and  
30 figure Figure 5 in the aforesaid article by Coutts/Edwards are considered, it is

apparent, at the interfaces between the objects, some of which at the same time represent interfaces between the processes, that the communication between the objects is carried out both ~~by means of~~ via synchronous method calls and ~~by means of~~ via interprocess communication in the form of the asynchronous passing on of messages. Such a definition of the method of communication at the interfaces of objects has the disadvantage that it is made considerably more difficult to re-use and maintain the objects.

~~In particular~~ Particularly in the context of the communication between various objects of a distributed system, also referred to as instances, which as a rule have what is known as a client/server relationship with one another and which are assigned to various processes, the procedure explained above is a very unfavorable solution in terms of the possibilities of re-use and maintenance which are desired in such a complex system.

A method for converting an interface definition description in an inter-object communications system is already known (EP-A-0 860 776) in which a client object and a server object are provided which are either operated on the same computer or on different computers.

The respective known method is based here on the function of, on the one hand, providing a programming method which is concentrated on origin processing, made available by a server object, while the advantages of a CORBA architecture, that is to say an architecture with a common (object request) broker are to be retained or ensured, and on the other hand of making available an inter-object communications method.

For this purpose there is provision to transmit a message from the client object to the server object in order to execute a specific processing operation. An interface definition conversion program here converts an interface definition description which is written in an interface definition language in order to generate what is referred to as a client stub, a server skeleton and a routing ~~program program~~ . The aforementioned interface definition description ~~comprises~~ includes one or more method definition descriptions of data which are necessary for the aforesaid specific

processing operation, and a processing result and a message description which specifies a format of the message which is to be output in response to the respective method definition description.

5 The aforementioned client stub is called in order to cause the client object to output the message. The server skeleton ~~comprises~~ includes a server registration method for storing a routing information item in a routing information memory in order to specify the format of the respective message which can process the server object itself when it is started. In addition, the aforementioned server skeleton ~~comprises~~ includes a method for describing the aforementioned specific processing  
10 operation which is to be executed when the server object receives the message.

Finally, the aforementioned routing program decides whether or not the processing of the server object assigned to the routing information item of the aforementioned message is possible, specifically by comparing the aforementioned message with the aforementioned routing information.  
15 ~~The method according to the present invention cannot be compared with this known procedure.~~

It is also known (EP-A-0 623 876) to allow object managers, which are provided on different computer platforms, to communicate with one another in a cooperative fashion while the objects are enabled to communicate on the respective  
20 computer platforms using a remote procedure request. ~~This procedure also has nothing to do with the method according to the invention.~~

Furthermore, a method and a device for carrying out efficient CORBA transactions are known (EP-A-0 834 807). However, in the method according to the present invention, procedures are not performed using CORBA transactions.

25 Finally, further methods for setting up connections between a server and a client are also known (US-A-5 802 367, WO 98 02814 A, GB-A-2 305 270); which, however, solve problems other than those solved by the present invention.

The An object of the present invention consists, therefore, in configuring a method for transmitting messages between what is referred to as client and server  
30 instances of a distributed system which are respectively assigned to different

processes, to the effect that in terms of the implementation of the method, the greatest possible degree of re-use is provided and, at the same time, maintenance is made as easy as possible.

### SUMMARY OF THE INVENTION

5           The object specified above is achieved according to the present invention with a method of the type mentioned at the beginning by virtue of the fact that, after reception of a message directed to at least one server instance by the client instance, a first instance (object handler 1), of partner instances provided as mutual communications partners, which contains the first process selects at least one  
10   suitable further instance(object handler 2), containing the at least one further process, of the partner instances, to receive and pass on messages with reference to an allocation table, and by virtue of the fact that the respective further instance (object handler 2) passes on the message to at least one server instance addressed by it; and, if appropriate, receives from the at least one server instance a message to be  
15   passed on to the client instance via the first instance containing the first process.

          The present invention provides the advantage that the definition of the method of communication between the client instance and the at least one server instance is exported into the partner instances which contain a process and which are provided as mutual communications partners. In this way, the messages between  
20   the client instance and the first instance containing the first process as well as between the further instance containing at least one server instance and the at least one further process are transmitted synchronously; for example, ~~by means of~~ via a procedure call or method call. The transmission of messages between a first instance containing the first process and a further instance containing at least one  
25   further process can then take place asynchronously or synchronously, in a way which is decoupled from the communications interfaces of the client instance and the at least one server instance. As a result, a maximum degree of re-use is achieved especially in terms of the implementation of the client instance and of the at least one server instance. The possibility of maintenance is also considerably improved  
30   by virtue of the fact that most communications interfaces between the first instance



containing the first process and the further instance containing the at least one further process have to be adapted, but the communications interfaces of the client and of the at least one server instance remain untouched.

5 A further advantageous ~~refinement~~ embodiment of the present invention provides for the allocation table to contain the type of messages which can be output by the client instance and the address of the further instance containing at least one further process. The type of messages which can be output by the client instance and the address of the further instance which contains at least one further process are therefore entered in this allocation table. This allocation table has the  
10 advantage that its contents can be changed at any time and that it is made possible for the first instance containing the first process to make a rapid selection.

According to one ~~expedient development~~ embodiment of the present invention, the selection made by the first instance containing the first process can be modified dynamically as a function of the system loading. As a result, the system  
15 crashes and blockages in the allocation of the processes to the processors can be avoided.

A further ~~refinement~~ embodiment of the present invention relates to the special case in which the first process and the at least one further process coincide. In this case, the first instance containing the first process and the further instance  
20 containing at least one further process are combined in one instance. As a result, the method according to the present invention can be applied to this special case without adaptation.

A further useful ~~refinement~~ embodiment of the present invention can be seen in the method of implementation. For example, all the instances (client  
25 instance, server instance, the instance containing the first process and the partner instance) can be implemented in the form of objects whose structure is defined by object classes. Thus, the first instance containing the first process and the further instance containing the at least one further process preferably each having the structure of a common object class. In this way, the principles of purely

object-oriented programming are utilized, permitting a high degree of modularity and of re-use and ease of maintenance.

A further ~~refinement~~ embodiment of the present invention is the very expedient use of the method ~~according to the invention~~ in a telephone switching system. According to this, all the advantages mentioned above ~~can~~ also can be exploited in conjunction with a telephone switching system.

Additional features and advantages of the present invention are described in, and will be apparent from, the following Detailed Description of the Invention and the Figures.

10 ~~An exemplary embodiment of the invention is described in more detail below with reference to a drawing, in which:~~

#### BRIEF DESCRIPTION OF THE FIGURES

Figure 1 shows an exemplary flowchart of the method according to the present invention.

15 Figure 2 shows an example of an application in the field of a system alarm in a telecommunications system such as a telephone switching system.

A key to the figures is provided at the end of the Detailed Description.

~~A key to the figures is provided in an annex at the end of the description.~~

#### DETAILED DESCRIPTION OF THE INVENTION

20 Figure 1 describes, in a flowchart, the transmission of messages between a client instance assigned to a first process and a server instance assigned to a further process. The instances of client, server, the first instance containing the first process and the further instance containing the at least one further process as well as the action which is carried out by the server instance are represented in the form of  
25 objects with boxes. Accordingly, the object client corresponds to a client instance, the object server to a server instance, the object object handler 1 to a first active instance containing the first process, of the partner instances provided as mutual communications partners, the object object handler 2 to a further active instance containing the further process, of the partner instances, the object action to an  
30 action and the object confirm action to an acknowledgement action to a requested

action. The active instances which contain the respective processes are characterized here ~~by means of~~ via boxes with bold lines. The type of action is not determined until the specific object action is called.

5 In the case of an action which is requested by the client and is to be carried out by the server, with an acknowledgement, the method proceeds, for example, as follows:

The client requests from the server an action to which an acknowledgement is to be made. The client calls the action and does not need to know which process is to be carried out or on which processor platform the action is to be carried out.

10 The object handler 1 provides the client for this purpose with the call procedure `invoke_action`. After the call of the procedure `invoke_action`, also referred to as method in the object-oriented programming, a uniquely defined number (get handle number) is assigned to the object handler 1 and a timer is started (start timer) which initiates error handling if the acknowledgement is not received at the correct time.

15 Then, the object handler 1 looks for a partner instance provided as communications partner, for example object handler 2 (find target object handler), which is assigned to the action as a function of the type of action, and transmits the message of the action request `action_request` to the object handler 2. The object handler 2 receives the message, stores the address of its communications partner object handler 1 (store communication partner) together with the number which is uniquely assigned to the object handler 1 and executes the procedure of the object action (`execute`).

20 The object action subsequently causes the server addressed by the client to execute the action ~~by means of~~ via the procedure call action. After the execution of the action, the server transmits, in an analogous fashion, an acknowledgement indirectly back to the client. According to this, the following procedure calls, message transmissions and actions run from the server in the direction of the client.

The procedure call `invoke_action`, deletes the address of the communications partner and transmits the action request message for the acknowledgement `action_request` of the object handler 2 to the object handler 1, which is known to the  
30 object handler 2 on the basis of the assigned number, the object handler 1 deletes

the assigned number (release handle number) and stops the timer (stop timer), in order to transmit the acknowledgement object handler 1 calls the procedure execute of the object Confirm action, and finally Object Confirm Action executes the procedure confirm\_action of the client.

5           In the case of an action of the server requested by the client being without an acknowledgement, the method according to the present invention of the transmission of messages from the client to the server runs in a similar fashion to that described above. The sequence steps get handle number, start timer, store communication partner and the steps relating to the acknowledgement from the  
10       server in the direction of the client are eliminated.

          In the case of what is referred to as a broadcast, i.e. a client requests an action from a plurality number of servers, there are various possibilities:

- if the servers addressed by the client are assigned to a common process, the object handler 1 will pass on the action\_request message either to an object  
15       handler 2, and the object handler 2 ensures that the action is executed by a plurality number of servers, or the object handler 1 transmits a plurality number of action\_request messages to a plurality number of object handlers 2 containing the server process, which each cause the servers to execute the action. A combination of both aforesaid variants is also possible.
- 20       -       if the servers addressed by the client are assigned to different processes, the object handler 1 will in each case transmit an action\_request message to the object handlers 2 containing the different processes, and the object handlers 2 in each case cause the servers to execute the action.

          Here too, all combinations of the aforesaid possibilities are conceivable.

25       The plurality number of actions usually have to be carried out in a distributed system so that ~~of course~~ each server ~~can~~ also can act as a client and each client ~~can~~ also can act as a server, and can be combined in an object client and server function.

          The advantageous decoupling of process interfaces from the object  
30       interfaces of the client and of the server is apparent from the fact that the

communication between the client and the server is implemented synchronously by means of via procedure calls and method calls and only the passing on of messages between the object handler 1 and the object handler 2 is, if appropriate, carried out asynchronously without limitation to the process limits.

5           In the special case in which the client and the server<sub>1</sub> which are located, for example, on a common platform, can be assigned to the same process, the objects object handler 1 and object handler 2 are combined to form a single object. According to Figure 1, ~~in this case~~ the object handler 1 transmits the action\_request message to itself.

10           Figure 2 shows an application example in the field of a system alarm in a telecommunications system<sub>2</sub>; for example, a telephone switching system.

In a system alarm, there are, for example, the following objects which can act simultaneously as client and server and can request different actions from one another. In addition, the objects can be located on different platforms.

15           An object Alarm Balance Monitor (ABM) has the function of forming an alarm balance over all the alarms of the instances (AMOI) which are monitored by it and for which alarms can be given. In order to be able to form the alarm balance, the Alarm Balance Monitor requires what is referred to as a SIBS object which is located on a processor platform and provides it with collected information relating  
20   to the monitored instances for which alarms can be given.

The boxes constitute the objects Caller, AMOI (AlarmManagedObjectInstance), SIBS (SiteBalanceSupply) and ABM (AlarmBalanceMonitor). The arrows whose type is not given in the key in the index indicate the message transmission, if appropriate, without limitation to process  
25   boundaries, between the objects. The transmission of messages correspond here to the transmission of messages between client and server described in Figure 1. Thus, for example, the Caller object can act as a client and the AMOI object as a server. The same also applies to the other objects AMOI and SIBS as well as SIBS and ABM.

After a system alarm call set\_alarm, the following sequence of actions is triggered, for example:

- Set\_alarm: a monitored instance AMOI for which an alarm can be given receives a new alarm from a caller, checks the parameters (check\_params) determining the alarm and creates a new alarm instance (create contained alarm).
- Confirm: an acknowledgement from the instance (AMOI) to the instance Caller after the system alarm call set\_alarm.
- Balance SIBS: at least one server object SIBS is requested to collect the received information necessary for the alarm balance (accumulate alarm status of all associated AMOI).
- Balance ABM: after this the server object ABM is requested to collect the information received from the at least one SIBS object for the alarm balance (accumulate alarm status of all associated SIBS).

Because the actions are requested without limitation to process boundaries, the messages are transmitted from one object to a further object via an active first instance and via a further active partner instance, for example via the object handler 1 and via the object handler 2 from Figure 1, neither of which is illustrated in Figure 2.

The selection of the object handler 2 made by the object handler 1 can be performed by means of via an allocation table. The allocation table looks, for example, as follows:

| Action         | Object handler   | Confirmation |
|----------------|------------------|--------------|
| Set_alarm AMOI | on AMOI platform | yes          |
| Balance SIBS   | on SIBS platform | no           |
| Balance ABM    | on main platform | no           |

If a specific action can be executed by different server objects, the allocation of the object handler 2 can be modified as a function of the system load factor.

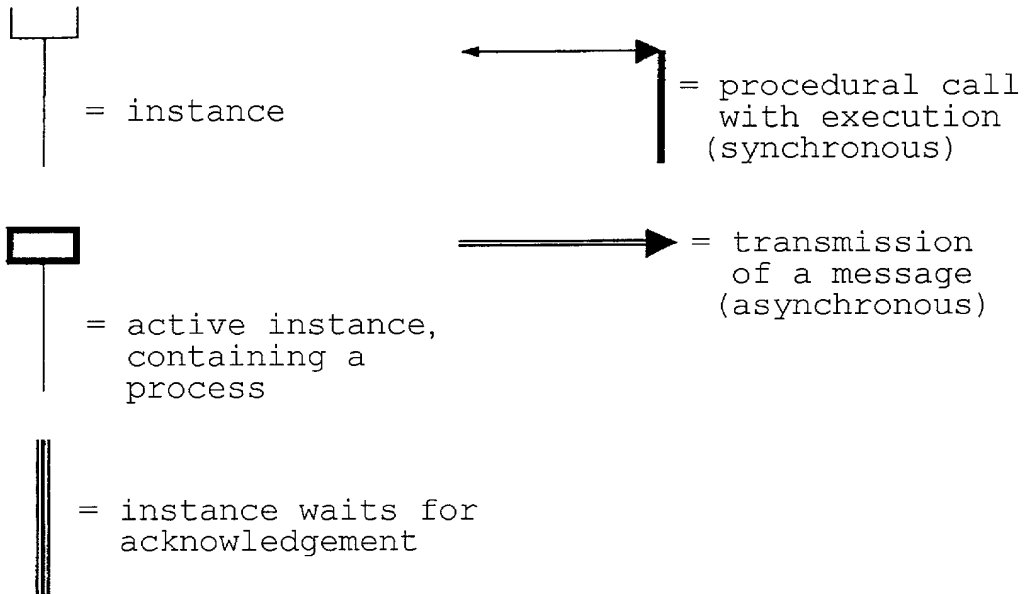
Although the present invention has been described with reference to specific embodiments, those of skill in the art will recognize that changes may be made

thereto without departing from the spirit and scope of the invention as set forth in  
the hereafter appended claims.

## APPENDIX

Key to the figures:

5





## Abstract

### ABSTRACT OF THE DISCLOSURE

Method for transmitting messages between a client instance assigned to a first process and at least one server instance assigned to at least one further process within a distributed system

- 5        A method for transmitting messages between a client instance assigned to a first process and at least one server instance assigned to at least one further process within a distributed system, wherein a first instance (object handler 1) containing a first process, of partner instances provided as mutual communications
- 10      partners, selects, after reception of a message directed to at least one server instance (server) by the client instance (client) at least one suitable further instance (object handler 2) containing the at least one further process, of the partner instances for the reception and passing on of messages. The at least one further instance containing the at least one further process passes on this message to at least one server instance
- 15      addressed by it and receives, if appropriate, from the at least server instance, a message to be passed on to the client instance via the first instance containing the first process.

Figure 1

1/2

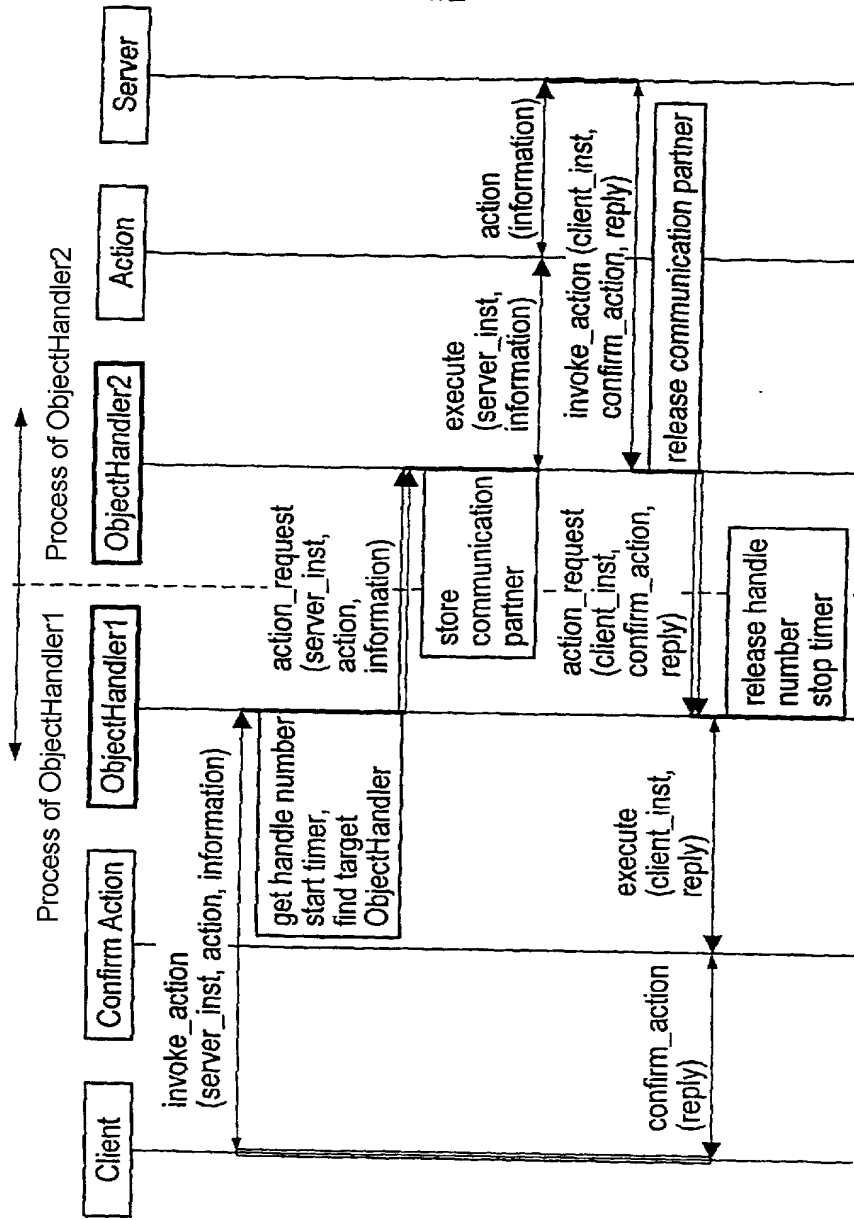
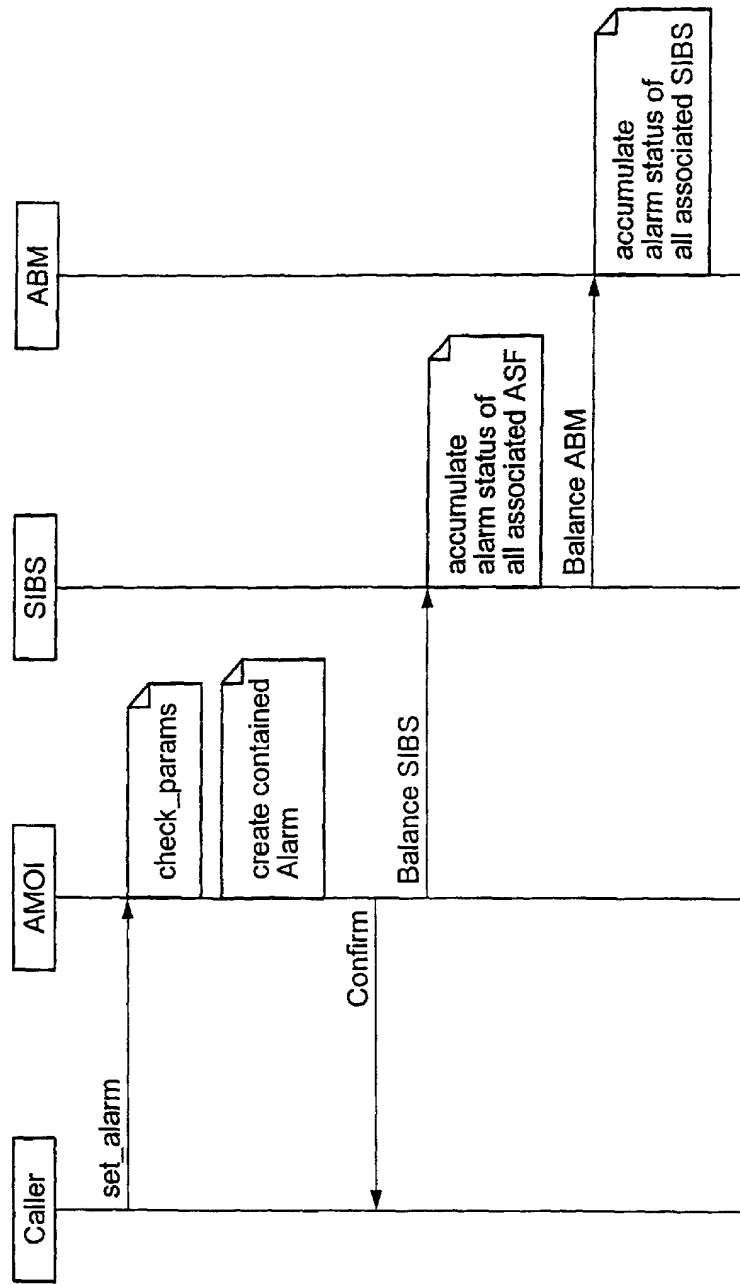


FIG 1



# Declaration and Power of Attorney For Patent Application

## *Erklärung Für Patentanmeldungen Mit Vollmacht*

### German Language Declaration

Als nachstehend benannter Erfinder erkläre ich hiermit an Eides Statt:

As a below named inventor, I hereby declare that:

dass mein Wohnsitz, meine Postanschrift, und meine Staatsangehörigkeit den im Nachstehenden nach meinem Namen aufgeführten Angaben entsprechen,

My residence, post office address and citizenship are as stated below next to my name,

dass ich, nach bestem Wissen der ursprüngliche, erste und alleinige Erfinder (falls nachstehend nur ein Name angegeben ist) oder ein ursprünglicher, erster und Miterfinder (falls nachstehend mehrere Namen aufgeführt sind) des Gegenstandes bin, für den dieser Antrag gestellt wird und für den ein Patent beantragt wird für die Erfindung mit dem Titel:

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

Verfahren zur Nachrichtenübertragung zwischen einer einem ersten Prozess zugewiesenen Clientinstanz und wenigstens einer mindestens einem weiteren Prozess zugewiesenen Serverinstanz innerhalb eines verteilten Systems

Method for transmitting, inside a distributed system, messages between a client instance assigned to a first process and at least one server instance assigned to at least one additional process

deren Beschreibung

the specification of which

(zutreffendes ankreuzen)

☐ hier beigefügt ist.

☒ am 01.03.2000 als

PCT internationale Anmeldung

PCT Anwendungsnummer PCT/DE00/00623

eingereicht wurde und am \_\_\_\_\_

abgeändert wurde (falls tatsächlich abgeändert).

(check one)

☐ is attached hereto.

☒ was filed on 01.03.2000 as

PCT international application

PCT Application No. PCT/DE00/00623

and was amended on \_\_\_\_\_

(if applicable)

Ich bestätige hiermit, dass ich den Inhalt der obigen Patentanmeldung einschliesslich der Ansprüche durchgesehen und verstanden habe, die eventuell durch einen Zusatzantrag wie oben erwähnt abgeändert wurde.

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims as amended by any amendment referred to above.

Ich erkenne meine Pflicht zur Offenbarung irgendwelcher Informationen, die für die Prüfung der vorliegenden Anmeldung in Einklang mit Absatz 37, Bundesgesetzbuch, Paragraph 1.56(a) von Wichtigkeit sind, an.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, §1.56(a).

Ich beanspruche hiermit ausländische Prioritätsvorteile gemäss Abschnitt 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 119 aller unten angegebenen Auslandsanmeldungen für ein Patent oder eine Erfindersurkunde, und habe auch alle Auslandsanmeldungen für ein Patent oder eine Erfindersurkunde nachstehend gekennzeichnet, die ein Anmeldedatum haben, das vor dem Anmeldedatum der Anmeldung liegt, für die Priorität beansprucht wird.

I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

PCT/DE00/00623

IDNR: 2590 / V: 99-1.00 / B: Val

# German Language Declaration

Prior foreign applications  
Priorität beansprucht

Priority Claimed

19910345.3

DE

09.03.1999

☒

☐

(Number)

(Country)

(Day Month Year Filed)

Yes

No

(Nummer)

(Land)

(Tag Monat Jahr eingereicht)

Ja

Nein

(Number)

(Country)

(Day Month Year Filed)

☐

☐

(Nummer)

(Land)

(Tag Monat Jahr eingereicht)

Yes

No

Ja

Nein

(Number)

(Country)

(Day Month Year Filed)

☐

☐

(Nummer)

(Land)

(Tag Monat Jahr eingereicht)

Yes

No

Ja

Nein

Ich beanspruche hiermit gemäss Absatz 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 120, den Vorzug aller unten aufgeführten Anmeldungen und falls der Gegenstand aus jedem Anspruch dieser Anmeldung nicht in einer früheren amerikanischen Patentanmeldung laut dem ersten Paragraphen des Absatzes 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 122 offenbart ist, erkenne ich gemäss Absatz 37, Bundesgesetzbuch, Paragraph 1.56(a) meine Pflicht zur Offenbarung von Informationen an, die zwischen dem Anmeldedatum der früheren Anmeldung und dem nationalen oder PCT internationalen Anmeldedatum dieser Anmeldung bekannt geworden sind.

I hereby claim the benefit under Title 35, United States Code, §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §122, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, §1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application.

PCT/DE00/00623

01.03.2000

anhängig

pending

(Application Serial No.)  
(Anmeldeseriennummer)

(Filing Date D, M, Y)  
(Anmeldedatum T, M, J)

(Status)  
(patentiert, anhängig,  
aufgegeben)

(Status)  
(patented, pending,  
abandoned)

(Application Serial No.)  
(Anmeldeseriennummer)

(Filing Date D,M,Y)  
(Anmeldedatum T, M; J)

(Status)  
(patentiert, anhängig,  
aufgeben)

(Status)  
(patented, pending,  
abandoned)

Ich erkläre hiermit, dass alle von mir in der vorliegenden Erklärung gemachten Angaben nach meinem besten Wissen und Gewissen der vollen Wahrheit entsprechen, und dass ich diese eidesstattliche Erklärung in Kenntnis dessen abgebe, dass wissentlich und vorsätzlich falsche Angaben gemäss Paragraph 1001, Absatz 18 der Zivilprozessordnung der Vereinigten Staaten von Amerika mit Geldstrafe belegt und/oder Gefängnis bestraft werden können, und dass derartig wissentlich und vorsätzlich falsche Angaben die Gültigkeit der vorliegenden Patentanmeldung oder eines darauf erteilten Patentes gefährden können.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

# German Language Declaration

VERTRETUNGSVOLLMACHT: Als benannter Erfinder beauftrage ich hiermit den nachstehend benannten Patentanwalt (oder die nachstehend benannten Patentanwälte) und/oder Patent-Agenten mit der Verfolgung der vorliegenden Patentanmeldung sowie mit der Abwicklung aller damit verbundenen Geschäfte vor dem Patent- und Warenzeichenamt: (Name und Registrationsnummer anführen)

POWER OF ATTORNEY: As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith. (list name and registration number)



29177

Customer No. 29177

And I hereby appoint

PATENT TRADEMARK OFFICE

Telefongespräche bitte richten an:  
(Name und Telefonnummer)

Direct Telephone Calls to: (name and telephone number)

Ext. \_\_\_\_\_

Postanschrift:

Send Correspondence to:

Bell, Boyd & Lloyd LLC  
Three First National Plaza, 70 West Madison Street, Suite 3300 60602-4207 Chicago, Illinois  
Telephone: (001) 312 372 11 21 and Facsimile (001) 312 372 20 98

or

Customer No. 29177

|                                                                                      |                         |                                                                   |      |
|--------------------------------------------------------------------------------------|-------------------------|-------------------------------------------------------------------|------|
| Voller Name des einzigen oder ursprünglichen Erfinders:<br><b>Dr. MICHAEL WAGNER</b> |                         | Full name of sole or first inventor:<br><b>Dr. MICHAEL WAGNER</b> |      |
| Unterschrift des Erfinders<br><i>Michael Wagner</i>                                  | Datum<br><b>9.12.01</b> | Inventor's signature                                              | Date |
| Wohnsitz<br><b>MUENCHEN, DEUTSCHLAND</b>                                             |                         | Residence<br><b>MUENCHEN, GERMANY</b>                             |      |
| Staatsangehörigkeit<br><b>DE</b> <i>DEU</i>                                          |                         | Citizenship<br><b>DE</b>                                          |      |
| Postanschrift<br><b>GLEIWITZER STR. 28</b>                                           |                         | Post Office Address<br><b>GLEIWITZER STR. 28</b>                  |      |
| <b>81929 MUENCHEN</b>                                                                |                         | <b>81929 MUENCHEN</b>                                             |      |
| Voller Name des zweiten Miterfinders (falls zutreffend):                             |                         | Full name of second joint inventor, if any:                       |      |
| Unterschrift des Erfinders                                                           | Datum                   | Second Inventor's signature                                       | Date |
| Wohnsitz                                                                             |                         | Residence                                                         |      |
| Staatsangehörigkeit                                                                  |                         | Citizenship                                                       |      |
| Postanschrift                                                                        |                         | Post Office Address                                               |      |
|                                                                                      |                         |                                                                   |      |

(Bitte entsprechende Informationen und Unterschriften im Falle von dritten und weiteren Miterfindern angeben).

(Supply similar information and signature for third and subsequent joint inventors).

0996636 1604